Date⇒ 04-01-2021
Module⇒ Backend
Lecture By⇒ Akash Handa
Subject ⇒ Revisiting Crud

IN PREVIOUS LECTURE **(QUICK RECAP)** **Date-02/02/2021**	**In Today's Lecture (Overview)**
Update command in mongodb Delete command in mongodb Upsert in Mongodb Crud Operation in Mongodb Create Operations Read Operations Update Operations Delete Operations	What is Crud Create Read Update Delete What is Cors What is Body parser What is Bson and What is the difference between Bson And Json How To Connect Nodejs With Mongodb

What is Crud

CRUD is an acronym that comes from the world of computer programming and refers to the four functions that are considered necessary to implement a persistent storage application: create, read, update and delete. Persistent storage refers to any data storage device that retains power after the device is powered off, such as a hard disk or a solid-state drive. In contrast, random access memory and internal caching are two examples of volatile memory - they contain data that will be erased when they lose power.

Create

The create function allows users to create a new record in the database. In the SQL relational database application, the Create function is called INSERT. In Oracle HCM Cloud, it is called create. Remember that a record is a row and that columns are termed attributes. A user can create a new row and populate it with data that corresponds to each attribute, but only an administrator might be able to add new attributes to the table itself.

Read

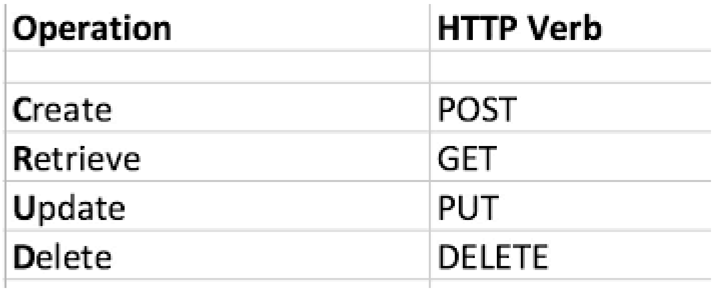
The read function is similar to a search function. It allows users to search and retrieve specific records in the table and read their values. Users may be able to find desired records using keywords, or by filtering the data based on customized criteria. For example, a database of cars might enable users to type in "1996 Toyota Corolla", or it might provide options to filter search results by make, model and year.

Update

The update function is used to modify existing records that exist in the database. To fully change a record, users may have to modify information in multiple fields. For example, a restaurant that stores recipes for menu items in a database might have a table whose attributes are "dish", "cooking time", "cost" and "price". One day, the chef decides to replace an ingredient in the dish with something different. As a result, the existing record in the database must be changed and all of the attribute values changed to reflect the characteristics of the new dish. In both SQL and Oracle HCM cloud, the update function is simply called "Update".

Delete

The delete function allows users to remove records from a database that is no longer needed. Both SQL and Oracle HCM Cloud have a delete function that allows users to delete one or more records from the database. Some relational database applications may permit users to perform either a hard delete or a soft delete. A hard delete permanently removes records from the database, while a soft delete might simply update the status of a row to indicate that it has been deleted while leaving the data present and intact

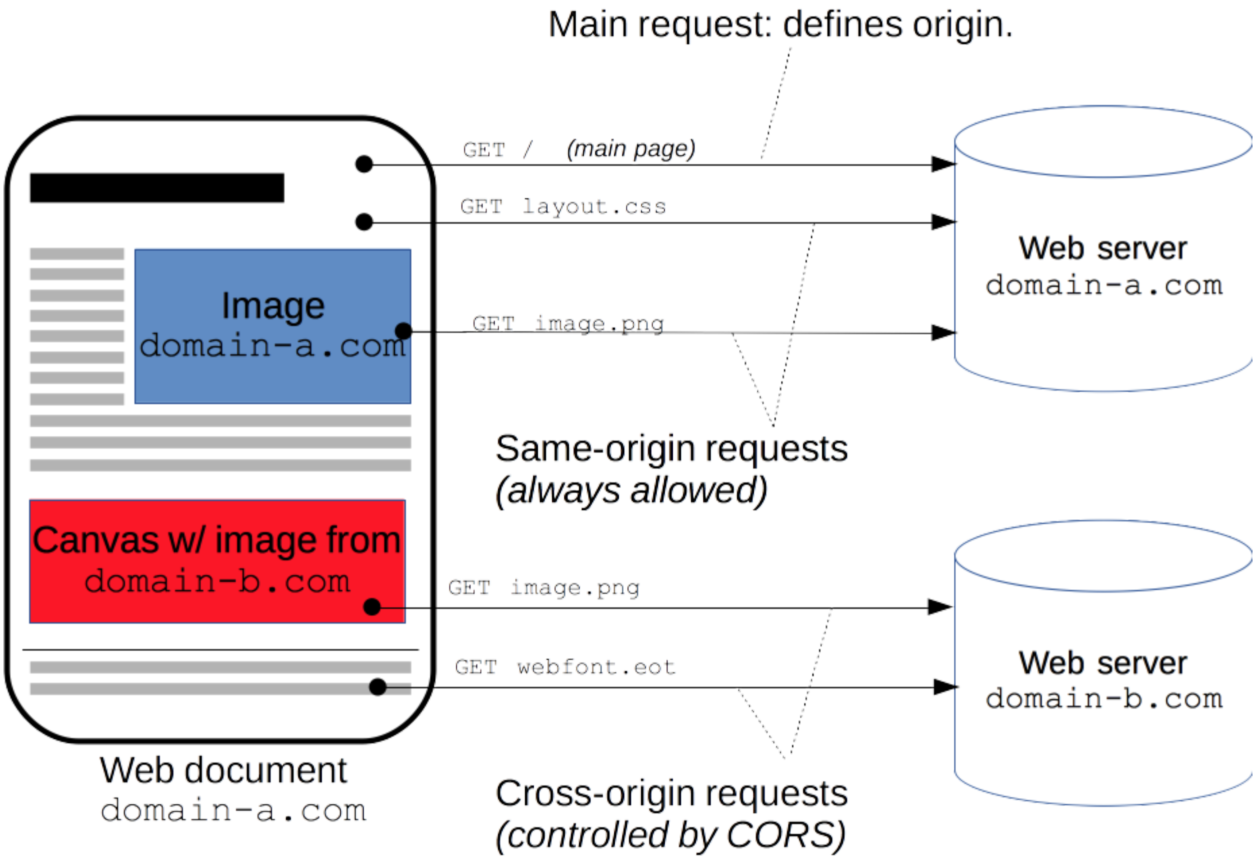


Operation	HTTP Verb
Create	POST
Retrieve	GET
Update	PUT
Delete	DELETE

What is Cors

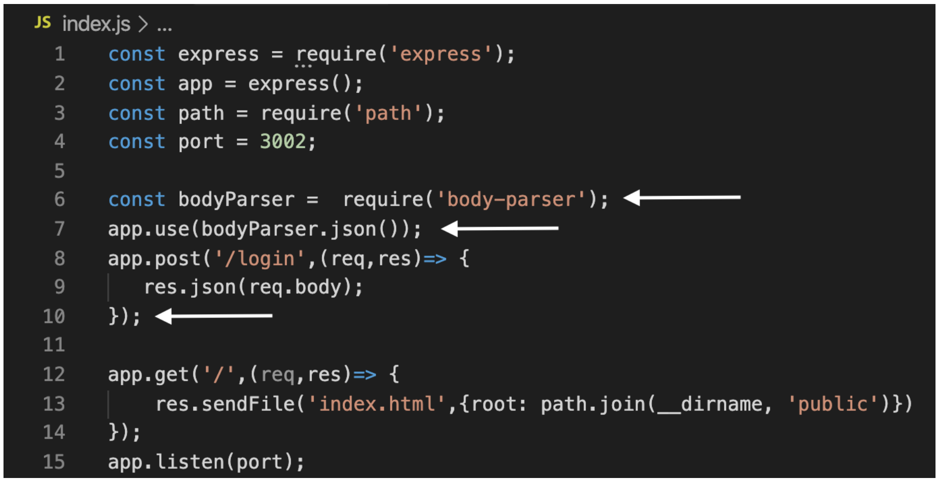
Cross-origin resource sharing is a mechanism that allows restricted resources on a web page to be requested from another domain outside the domain from which the first resource was served. A web page may freely embed cross-origin images, stylesheets, scripts, iframes, and videos.

Cross-Origin Resource Sharing (**CORS**) is an HTTP-header based mechanism that allows a server to indicate any other origins (domain, scheme, or port) than its own from which a browser **should** permit loading of resources. ... For security reasons, browsers restrict cross-origin HTTP requests initiated from scripts.



What is Body parser

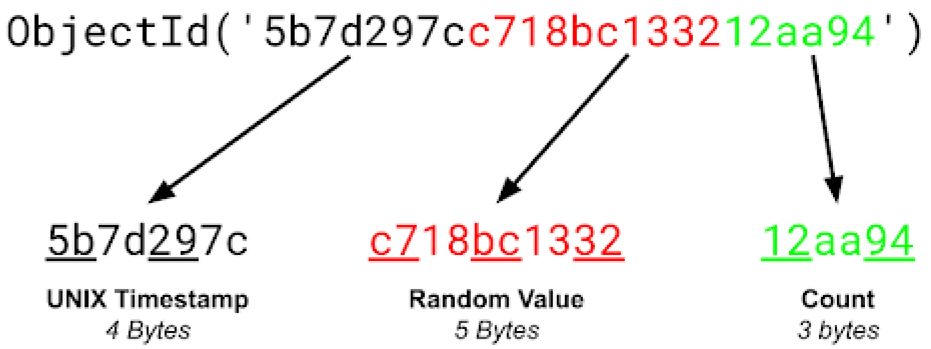
Body-parser is the Node. js **body parsing** middleware. It is responsible for **parsing** the incoming request bodies in a middleware before you handle it. Installation of **body-parser** module: You can visit the link to Install **body-parser** module.



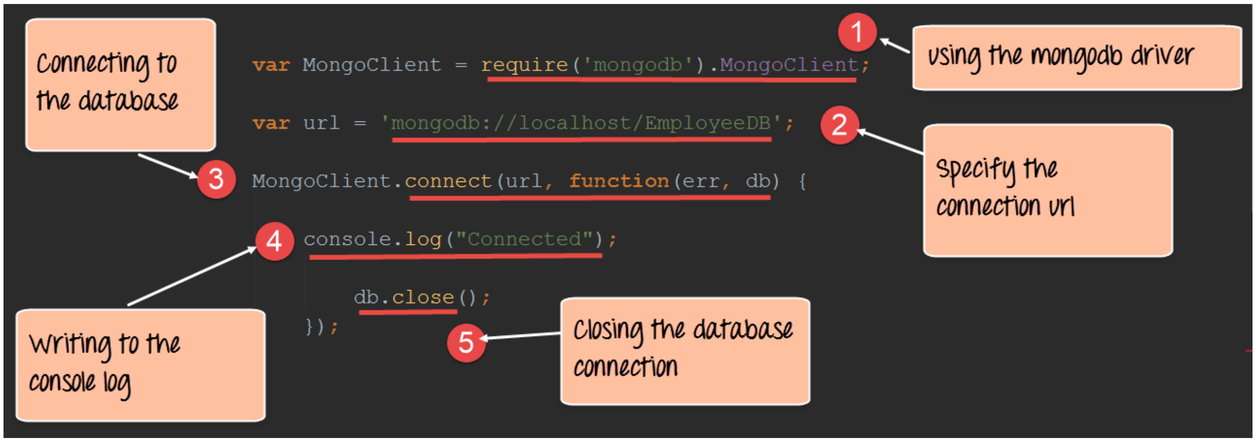
```js
const express = require('express');
const app = express();
const path = require('path');
const port = 3002;

const bodyParser =  require('body-parser');
app.use(bodyParser.json());
app.post('/login',(req,res)=> {
    res.json(req.body);
});

app.get('/',(req,res)=> {
    res.sendFile('index.html',{root: path.join(__dirname, 'public')})
});
app.listen(port);
```

What is Bson and What is the difference between Bson And Json

BSON is a computer data interchange format. The name "BSON" is based on the term JSON and stands for "Binary JSON". It is a binary form for representing simple or complex data structures including associative arrays, integer indexed arrays, and a suite of fundamental scalar types. BSON originated in 2009 at MongoDB.
BSON is a serialization format encoding format for **JSON** mainly used for storing and accessing the documents whereas **JSON** is a human-readable standard file format mainly used for transmission of data **in the** form of key-value attribute pairs. ... **BSON** in fact in some cases uses more space than **JSON**.

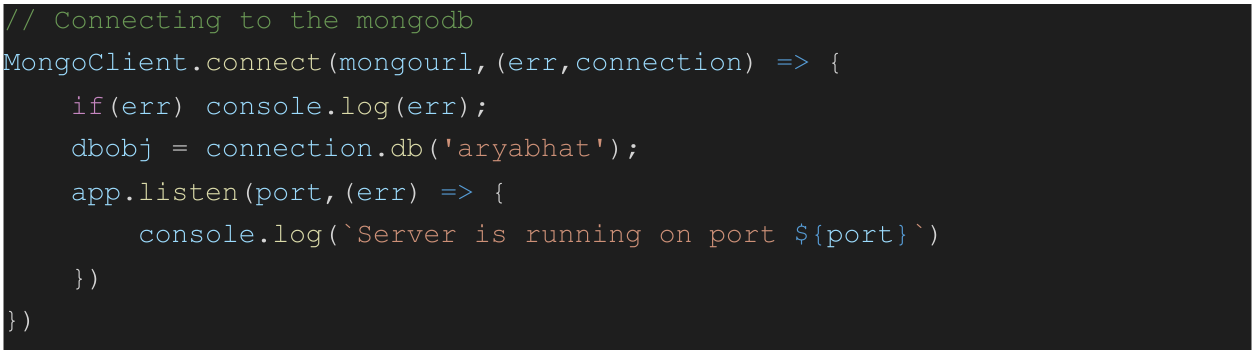


How To Connect Nodejs With Mongodb

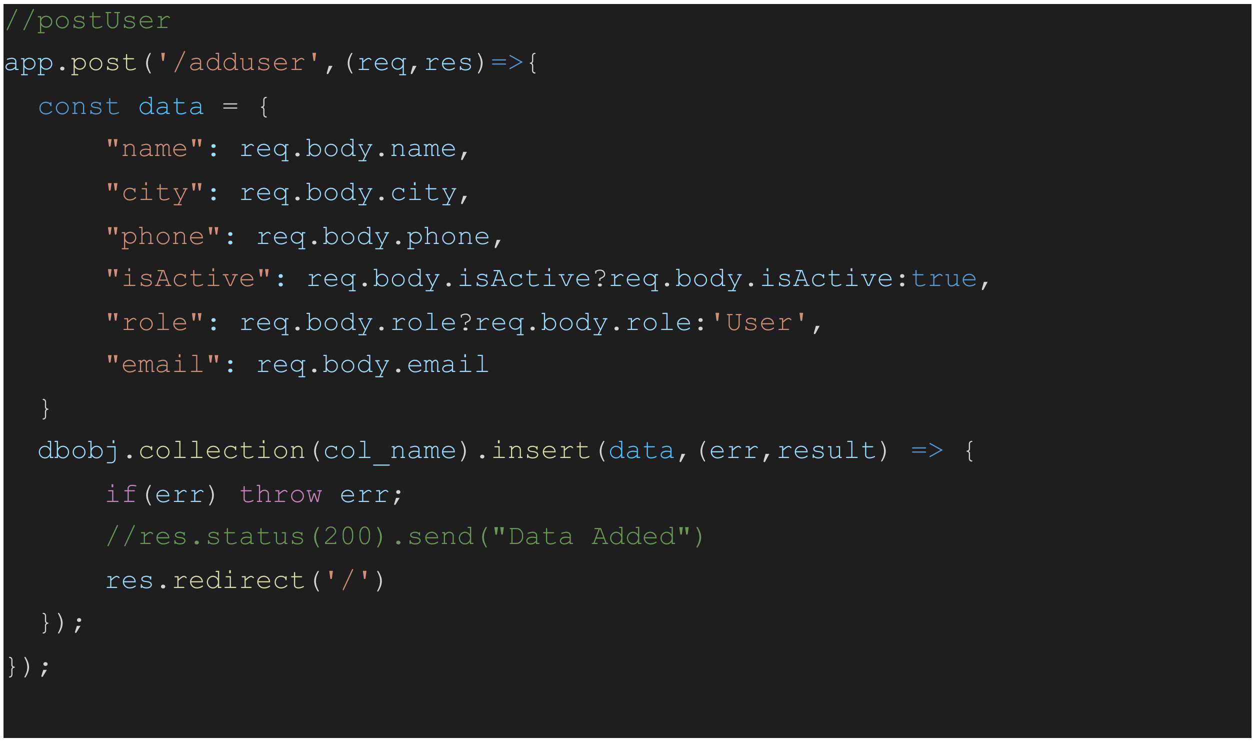


```
var MongoClient = require('mongodb').MongoClient;

var url = 'mongodb://localhost/EmployeeDB';

MongoClient.connect(url, function(err, db) {

    console.log("Connected");

        db.close();
    });
```

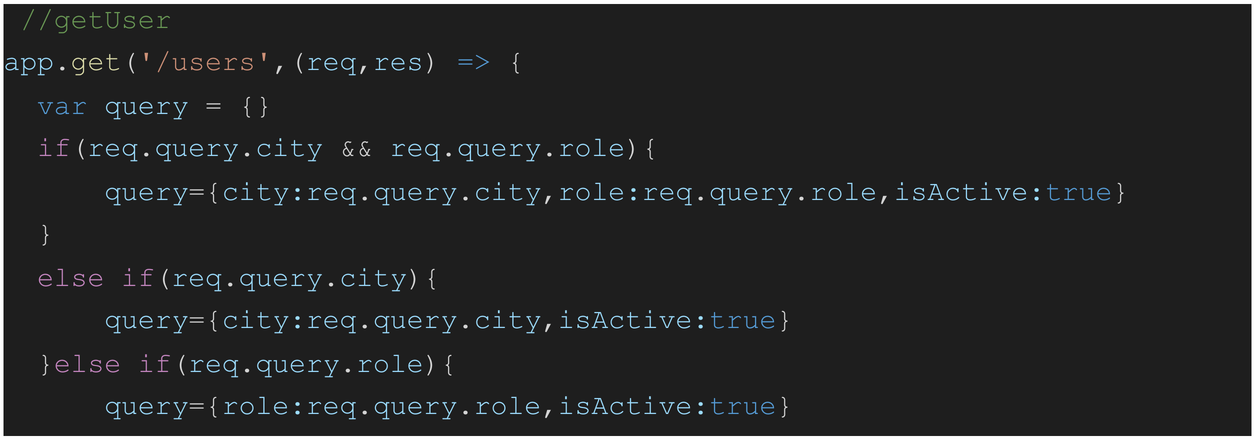
1
using the mongodb driver
2
Specify the connection url
Connecting to the database
3
4
Writing to the console log
5
Closing the database connection



```
// Connecting to the mongodb
MongoClient.connect(mongourl,(err,connection) => {
    if(err) console.log(err);
    dbobj = connection.db('aryabhat');
    app.listen(port,(err) => {
        console.log(`Server is running on port ${port}`)
    })
})
```



```
//postUser
app.post('/adduser',(req,res)=>{
  const data = {
      "name": req.body.name,
      "city": req.body.city,
      "phone": req.body.phone,
      "isActive": req.body.isActive?req.body.isActive:true,
      "role": req.body.role?req.body.role:'User',
      "email": req.body.email
  }
  dbobj.collection(col_name).insert(data,(err,result) => {
      if(err) throw err;
      //res.status(200).send("Data Added")
      res.redirect('/')
  });
});
```



```
 //getUser
app.get('/users',(req,res) => {
  var query = {}
  if(req.query.city && req.query.role){
      query={city:req.query.city,role:req.query.role,isActive:true}
  }
  else if(req.query.city){
      query={city:req.query.city,isActive:true}
  }else if(req.query.role){
      query={role:req.query.role,isActive:true}
  }else{
      query={isActive:true}
  }
  dbobj.collection(col_name).find(query).toArray((err,result) => {
      if(err) throw err;
      res.send(result)
  })
})
```

```
//UserDetails
app.get('/user/:id',(req,res) => {
  var id = mongo.ObjectID(req.params.id)
  var query = {}
  query={_id:id}
  dbobj.collection(col_name).findOne(query,(err,result) => {
      if(err) throw err;
      res.send(result)
  })
})
```